DOCUMENT RESUME

ED 243 453 IR 011 065

AUTHOR Post, Paul E.; Sarapin, Marvin I.
TITLE Writing and Evaluating Educational Software: Some Key Elements.
PUB DATE 3 Dec 83
NOTE 38p.; Paper presented at the Annual Conference of the American Vocational Association (Anaheim, CA, December 3, 1983). Printed on colored paper.
PUB TYPE Computer Programs (101) -- Speeches/Conference Papers (150)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Authoring Aids (Programing); *Computer Assisted Instruction; *Computer Software; Display Aids; Instructional Materials; Material Development; *Microcomputers; *Production Techniques; *Programing; Programing Languages
IDENTIFIERS *BASIC Programing Language; Computer Uses in Education

ABSTRACT

Following a brief description of types of computer assisted instruction (tutorial, drill and practice, and simulation/games), this document provides the following BASIC programming routines: variable typing, range error maskings, default entries, having user check input, allowing users to change input, response checker for numbers, response checker for strings, a simple parser, setup interactive style, interactive style, screen scroller, screen oriented format, window poker, print using for numbers, centering text, word wrap, putting text in boxes, printer interfacer, typewriter sound, menu maker, user self pacer, user pacer pre-defined, right answer routine, wrong answer routine, section feedback, fading prompts, help screens, and exit routine. Each description includes purpose and use of the routine, how to check a program for the routine, and how to program the routine in BASIC. (LMM)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED243453

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

- This document has been reproduced as received from the person or organization originating it.
- Minor changes have been made to improve reproduction quality.

- Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

American Vocational Association
1983 Annual Convention
Anaheim, California
December 3

Writing and Evaluating Educational Software: Some key elements

IR011065

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

M. Sarapin

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

Types of CAI

Computer assisted instruction is a generic term covering all uses of the computer for actual instruction. A major component of CAI has been the use of the computer to present material to, or provide practice for the student. The following information identifies the most common types of CAI being used in educational applications.

Tutorial

Tutorial CAI is characterized by the computer serving as a presentor of new information. There are two subtypes of tutorial CAI found in the literature.

Linear and Branching Tutorials are direct descendents of programmed instruction which use the computer as a teaching machine. A linear tutorial takes all students straight through the material to be learned. A branching tutorial allows students to traverse different routes through the material. Most often, tutorials use the computer as little more than an electric page turner.

Dialog tutorials are more interactive than linear and branching tutorials using "conversation" between the student and the computer. Dialog tutorials make more complete use of the computer's abilities than direct linear CAI. Dialogs are descendents of the socratic method of teaching using a questioning technique to instruct the student.

Drill and Practice

Drill and practice CAI provides opportunity for the student to use information previously obtained, much as flash cards are used. Drill and practice is used most often where rote memory is important. Drill and practice consists of solving a set of problems presented by the computer.

Simulation/Games

Simulation/games CAI not only allow the student to use information previously learned, but can also be used to implement discovery learning. These subtypes of simulation/games CAI have been identified in the literature.

Games involve using the computer to compete with students, i.e. the game may be won or lost. Games provide opportunity for students to practice skills.

Simulations of mechanical systems, such as a nuclear reactor or an automobile engine allow students to manipulate a physical environment which may not be otherwise practical.

Simulations of non-mechanical systems such as voters in an election or ecological systems allow students to study complex social and political environments.

There are no distinct dividing lines between the types of CAI. An individual lesson may in fact combine a number of CAI types. Within each CAI type there are often differences in interpretation and application.

Variable Typing

Description of purpose and use: A program written in BASIC expects input variables such as (A) to be presented with NUMBERS and variables such as (A$) to presented with alphanumeric characters (A-Z and 1-0). To avoid variable type errors in data input, many programmers treat all variables as $trings in INPUT commands. String responses can then be converted to the correct number by use of the VAL command. The Applesoft ASC command can also be used to convert the first letter of a string to decimal ASCII code.

How to check a program for routine: Most microcomputer versions of BASIC will give the user an error message if the wrong variable type is typed into the computer.

How to program routine in BASIC:

```
21100  REM  VARIABLE TYPING
21110  PRINT
21120  PRINT "PRESS 1 THEN 'RETURN' TO GO ON": PRINT
21130  FLASH : PRINT "TRY TO ENTER 'ONE'!";: NORMAL
21140  PRINT " ";
21150  INVERSE : PRINT "THEN TRY '1'!": NORMAL
21160  PRINT : INPUT "====>";A9: REM  SHOULD BE A9$
21170  PRINT : PRINT "TO AVOID ERRORS!"
21180  PRINT "USE $TRING VARIABLES FOR DATA INPUT!"
21190  PRINT : PRINT : PRINT :VP = 23: GOSUB 8000: HOME
```

Range Error Masking

Description of purpose and use: This technique screens out responses that are out of desired range.

How to check a program for routine: If a range screening or masking technique is not used a user can input undesired responses to a program. Out of range responses can cause many calculation problems or 'bomb' the program.

How to program routine in BASIC:

```
1000  REM  RANGE ERROR FEEDBACK
1010  REM  INPUT VARIABLES -
1020  REM  OUTPUT VARIABLES -
1030  REM  INTERNAL VARIABLES -  J9
1040  PRINT "THINK";
1050 TD = 2000: GOSUB 8500
1060  FOR J9 = 1 TO 5: PRINT  CHR$ (8);: REM  BACK CURSOR UP

1070  NEXT J9
1080  CALL  - 868: REM  CLEAR RIGHT/CURSOR
1090  RETURN
```

How to call routine from program: The desired range is checked by ASCII value in main program. Out of range responses branch program to subroutine. Subroutine gives the THINK feedback then resets the cursor for a new response.

Sample program usage:

```
21200  REM  RANGE ERROR
21210  HOME
21220  PRINT "ENTER A NUMBER FROM 1 TO 9": PRINT
21230  FLASH : PRINT "TRY TO ENTER '0' OR 'A'!";:
21240  NORMAL : PRINT " ";
21250  INVERSE : PRINT "THEN TRY '6'!": NORMAL : PRINT
21260  GET AN$
21270  IF  ASC (AN$) < 49 OR  ASC (AN$) > 57 THEN  GOSUB
1000: GOTO 21260
21280  PRINT :T$ = "IMPROVE PROGRAMMING BY MASKING OFF OUT
OF RANGE RESPONSES.": GOSUB 6300
21290 T$ = "STOP": GOSUB 6300:VP = 23: GOSUB 8000
```

Default Entries

Description of purpose and use: This routine allows the user to just press return in order to input a common response. This saves typing time and reduces the chance of error.

How to check a program for routine: If the program prompts the user with common responses requiring the user to only press RETURN (or ENTER) then the program has a default entry routine.

How to program routine in BASIC:

```
2000  REM  USING DEFAULT ENTRIES
2010  REM  INPUT VARIABLES - DE$ (DEFAULT)
2020  REM  OUTPUT VARIABLES - A$ (ANSWER)
2030  REM  INTERNAL VARIABLES - I9, A9$, A8$
2040  INVERSE
2050  PRINT DE$;
2060  GET A9$
2070  IF A9$ =  CHR$ (13) THEN A$ = DE$: NORMAL : RETURN
2080  FOR I9 = 1 TO  LEN (DE$): PRINT  CHR$ (8);: NEXT I9
2090  CALL  - 868: REM  CLEAR TO END OF LINE
2100  NORMAL
2110  PRINT A9$;
2120  INPUT "";A8$
2130 A$ = A9$ + A8$
2140  RETURN
```

How to call routine from program: The input variable DE$ (Default Entry) must be set in the main program before calling the routine. The output variable A$ (Answer $tring) returns the user's answer to the main program. If you want the answer to be a number use the statement A = VAL (A$) before returning to the main program.

Sample program usage:

```
22100  REM  DEFAULT ENTRIES
22110  PRINT "ARE DEFAULT ENTRIES USEFUL? ";
22120 DE$ = "Y": GOSUB 2000
22130  HOME
22140  PRINT : PRINT : PRINT "INSTEAD OF TYPING IN A
FREQUENTLY USED"
22150  PRINT : PRINT "VALUE, ERRORS CAN BE REDUCED BY
OFFERING";
22160  PRINT : PRINT "THE USER A - ";
22170 DE$ = "DEFAULT": GOSUB 2000
22180  IF A$ <  > "DEFAULT" THEN 22200
22190  GOTO 22300
22200  PRINT : PRINT "WRONG, DEFAULTS DO HELP REDUCE
ERRORS."
22210 VP = 21: GOSUB 8000
22220  GOTO 22190
```

Having User Check Input

Description of purpose and use: Allows user to check for typing mistakes. If a mistake is found the user may then go and correct it.

How to check a program for routine: A program has this routine if it asks the user if the input data is correct. If the data is incorrect the program should allow the user to change the data.

How to program routine in BASIC:

```
2200  REM  HAVING USER CHECK INPUT
2210  REM  INPUT VARIABLES -
2220  REM  OUTPUT VARIABLES - A1$
2230  REM  INTERNAL VARIABLES -
2240  VTAB 23
2250  PRINT "IS THIS INPUT CORRECT? ";: GET A1$
2260  VTAB 23: HTAB 1: CALL  - 868: REM  CLEAR TO END OF
LINE
2270  RETURN
```

How to call routine from program: This routine asks the user if the input is correct and returns the users answer to the main program as variable A1$. The main program should then branch according to A1$ as in line 22420 in the sample program.

Sample program usage:

```
22300  REM   CHECKING INPUT & CHANGING INPUT
22310  HOME
22320  PRINT "WHAT IS THE NAME OF YOUR ALMA MATER? ": PRINT
22330  INPUT "";A$
22340  GOSUB 2200
22350  IF A1$ = "N" THEN 22400
22360  HOME
22370  VTAB 10: PRINT "ANSWER THE 1ST QUESTION WRONG THIS
TIME"
22380 VP = 23: GOSUB 8000
22390  GOTO 22300
22400 V9 = 10:H9 = 1: GOSUB 2400
22410  GOSUB 2200
22420  IF A1$ = "N" THEN 22400
22430  VTAB 15
22440  PRINT "I HOPE ";A$;" IS HAVING"
22450  PRINT : PRINT "A GOOD YEAR IN SPORTS."
22460 VP = 23: GOSUB 8000
```

Allowing User To Change Input

Description of purpose and use: Allows user to correct incorrect input responses by retyping incorrect portions of the original response and copying correct portions by using the right arrow key.

How to check a program for routine: A program has some form of this routine if it allows the user to correct responses by retyping incorrect portions and copying correct responses portions. The cursor restarts at the original input location to facilitate the process.

How to program routine in BASIC:

```
2400  REM  ALLOWING USER TO CHANGE INPUT
2410  REM  INPUT VARIABLES - A$, V9, H9
2420  REM  OUTPUT VARIABLES -
2430  REM  INTERNAL VARIABLES - A9$
2440  VTAB V9: HTAB H9
2450  PRINT A$;
2460  VTAB V9: HTAB H9
2470  INPUT "";A9$
2480  IF A9$ = "" THEN  RETURN
2490 A$ = A9$
2500  RETURN
```

How to call routine from program: The main program must pass the original incorrect response A$ (Answer $tring). The desired cursor location is passed to the routine by V9 (Vertical location) and H9 (Horizontal location). The response A$ is set equal to the retyped response A9$ before program control is sent back to the main program.

Sample program usage:

```
22300  REM   CHECKING INPUT & CHANGING INPUT
22310  HOME
22320  PRINT "WHAT IS THE NAME OF YOUR ALMA MATER? ": PRINT
22330  INPUT "";A$
22340  GOSUB 2200
22350  IF A1$ = "N" THEN 22400
22360  HOME
22370  VTAB 10: PRINT "ANSWER THE 1ST QUESTION WRONG THIS
TIME"
22380 VP = 23: GOSUB 8000
22390  GOTO 22300
22400 V9 = 10:H9 = 1: GOSUB 2400
22410  GOSUB 2200
22420  IF A1$ = "N" THEN 22400
22430  VTAB 15
22440  PRINT "I HOPE ";A$;" IS HAVING"
22450  PRINT : PRINT "A GOOD YEAR IN SPORTS."
22460 VP = 23: GOSUB 8000
```

Response Checker For Numbers

Description of purpose and use: This routine allows user to answer numerical questions within a preset range (ie. + or - .001 of the desired response). Such a routine allows rounding errors in response checking.

How to check a program for routine: A program without a tolerance in a number input checking routine will not allow user rounding errors.

How to program routine in BASIC:

```
3000  REM  RESPONSE CHECKER FOR NUMBERS
3010  REM  INPUT VARIABLES - A, CA, TL
3020  REM  OUTPUT VARIABLES - CR
3030  REM  INTERNAL VARIABLES -
3040 CR = 0
3050  IF (A >  = CA - TL) AND (A <  = CA + TL) THEN CR = 1
3060  RETURN
```

How to call routine from program: The input variables: A (user Answer), CA (Correct Answer), and TL (ToLerance) must be passed from the main program. The output variable CR (CoRrect) will be returned as (1) if response is within the desired range; (0) is returned if response is out-of-range.

Sample program usage:

```
23100  REM  NUMBERS
23110  INPUT "WHAT IS 17 + 8? ";AS
23120  GOSUB 2200
23130  IF A1$ <  > "Y" THEN V9 = 8:H9 = 17: GOSUB 2400
23140 A =  VAL (A$)
23150 CA = 25:TL = .001: GOSUB 3000
23160  VTAB 10:VP = 23
23170  IF CR = 1 THEN  GOSUB 9000: GOSUB 8000
23180  IF CR = 0 THEN  GOSUB 9200: GOSUB 8000
```

Response Checker For Strings

Description of purpose and use: This routine allows program to check a user $tring response against a desired response. Any initial spaces typed by the user prior to the response are ignored.

How to check a program for routine: A string answer checking routine without facility for excluding initial blanks will keep the blank as part of the literal string. This can cause improper error checking.

How to program routine in BASIC:

```
3100  REM  RESPONSE CHECKER FOR STRINGS
3110  REM  INPUT VARIABLES - A$, CA$
3120  REM  OUTPUT VARIABLES - CR
3130  REM  INTERNAL VARIABLES -
3140 CR = 0
3150  IF  LEFT$ (A$,1) = " " THEN A$ =  RIGHT$ (A$,( LEN
(A$) - 1)): GOTO 3150
3160  IF A$ = CA$ THEN CR = 1
3170  RETURN
```

How to call routine from program: The routine requires the input variables A$ (the user Answer $tring) and CA$ (the Correct Answer $tring) to be passed from the main program. After the routine removes initial response spaces, A$ and CA$ are evaluated. The output variable CR (CoRrect) is returned as (1) if the strings are identical; (0) is returned if not.

Sample program usage:

```
23300  REM  STRINGS
23310  HOME
23320  PRINT "WHAT IS THE PART OF THE COMPUTER USED BY";:
PRINT
23330  PRINT "HUMANS TO GIVE INFORMATION TO COMPUTERS?";:
PRINT
23340  INPUT "";A$
23350  GOSUB 2200
23360  IF A1$ <  > "Y" THEN V9 = 12:H9 = 1: GOSUB 2400
23370 CA$ = "KEYBOARD": GOSUB 3100
23380  VTAB 15:VP = 23
23390  IF CR = 1 THEN  GOSUB 9000: GOSUB 8000
23400  IF CR = 0 THEN  GOSUB 9200: GOSUB 8000
```

A Simple Parser

Description of purpose and use: This parser routine allow user to type several words (sentence) in response to a question. The parser then looks at response for a desired word.

How to check a program for routine: Programs with a parser routine for response checking will look for significant portions of a user input.

How to program routine in BASIC:

```
3200  REM  A SIMPLE PARSER
3210  REM  INPUT VARIABLES - A$, CA$
3220  REM  OUTPUT VARIABLES - CR
3230  REM  INTERNAL VARIABLES - I9
3240 CR = 0
3250  FOR I9 = 1 TO (( LEN (A$) -  LEN (CA$)) + 1)
3260  IF CA$ =  MID$ (A$,I9,( LEN (CA$))) THEN CR = 1: GOTO
3280
3270  NEXT I9
3280  RETURN
```

How to call routine from program: This parser requires the input variables A$ (user Answer $tring) and CA$ (Correct Answer String) to be passed from the main program. The A$ is parsed to see if it includes CA$. The output variable CR (CoRrect) is set to (1) if A$ includes CA$; CR is set to (0) if not.

Sample program usage:

```
23500  REM  PARSING
23510  HOME
23520  PRINT "TYPE IN A SENTENCE NAMING THE TYPE OF"
23530  PRINT : PRINT "MEMORY LOST WHEN A COMPUTER LOSES
POWER."
23540  INPUT "";A$
23550  GOSUB 2200
23560  IF A1$ <  > "Y" THEN V9 = 12:H9 = 1: GOSUB 2400
23570 CA$ = "RAM": GOSUB 3200
23580  VTAB 15:VP = 23
23590  IF CR = 1 THEN  GOSUB 9000: GOSUB 8000
23600  IF CR = 0 THEN  GOSUB 9200: GOSUB 8000
```

Setup Interactive Style

Description of purpose and use: This routine asks user to input number of letters in first name (up to 20). User is then asked to type name (return key does not need to be pressed). The typed name can be used in the remainder of the program as an interactive prompt.

How to check a program for routine: If a program asks for the user's name for later use, some form of this routine is being used.

How to program routine in BASIC:

```
4000  REM  SET UP INTERACTIVE STYLE
4010  REM  INPUT VARIABLES -
4020  REM  OUTPUT VARIABLES - N$
4030  REM  INTERNAL VARIABLES - NL,NL$,N9$,I9
4040 N$ = ""
4050  PRINT "PLEASE TYPE THE NUMBER OF LETTERS IN"
4060  INPUT "YOUR FIRST NAME THEN PRESS 'RETURN'->";NL$
4070  LET NL =  VAL (NL$)
4080  IF NL < 1 OR NL > 20 THEN  HOME : GOTO 4000
4090  PRINT : PRINT "PLEASE TYPE YOUR FIRST NAME"
4100  PRINT "====>";: FOR I9 = 1 TO NL: PRINT ".";: NEXT I9
4110  FOR I9 = 1 TO NL: CALL  - 1008: NEXT I9: REM   BACK
CURSOR
4120  FOR I9 = 1 TO NL: GET N9$
4130  IF  ASC (N9$) = 8 THEN N$ =  LEFT$ (N$,( LEN (N$) -
1)): GOTO 4160
4140  IF N9$ =  CHR$ (13) THEN  RETURN
4150  LET N$ = N$ + N9$
4160  IF  ASC (N9$) = 8 THEN  PRINT N9$;".";:I9 = I9 - 1
4170  PRINT N9$;
4180  NEXT I9: RETURN
```

How to call routine from program: This routine can be called from any location in the main program. However, it is usually called early to facilitate use of the the user's name in prompts and responses. The routine returns the output variable N$ (user Name String).

Sample program usage:

```
24000  REM  INTERACTION STYLE
24010  HOME
24020 TS = "INTERACTION STYLE": GOSUB 6500
24030 WL = 0:WW = 40:WT = 7:WB = 24: GOSUB 5000
24040  GOSUB 4000: PRINT : PRINT : GOSUB 4200
24050 VP = 22: GOSUB 8000
```

Interactive Style

Description of purpose and use: Allows program to prompt user with his/her name during processing. The user's name must have first been identified in main program or in the Setup Interactive Style routine. Routine combines user's name with other comments on a random basis.

How to check a program for routine: If a program returns user's name and other prompts or comments on a random basis, some form of this routine is being used.

How to program routine in BASIC:

```
4200  REM  INTERACTIVE STYLE
4210  REM  INPUT VARIABLES - N$
4220  REM  OUTPUT VARIABLES -
4230  REM  INTERNAL VARIABLES - X9
4240  PRINT
4250 X9 =  INT ( RND (1) * 3) + 1
4260  ON X9 GOTO 4270,4280,4290
4270  PRINT "OK! ";N$;".": RETURN
4280  PRINT "LET'S KEEP GOING ";N$;"!": RETURN
4290  PRINT N$;", IT'S TIME TO MOVE ON!": RETURN
```

How to call routine from program: This routine can be called from any location in the main program after N$ (user's Name String) is identified. Routine prints response on screen after a random selector is generated.

Sample program usage:

```
24000  REM  INTERACTION STYLE
24010  HOME
24020 T$ = "INTERACTION STYLE": GOSUB 6500
24030 WL = 0:WW = 40:WT = 7:WB = 24: GOSUB 5000
24040  GOSUB 4000: PRINT : PRINT : GOSUB 4200
24050 VP = 22: GOSUB 8000
```

Screen Scroller

Description of purpose and use: This routine creates the visual effect of moving text information up on the screen one page or screen at a time (usually called screen scrolling). Each new PAGE is started at the bottom of the screen and scrolled up.

How to check a program for routine: A program without a page or screen orientation will scroll text lines one at a time with no visual starting or ending point.

How to program routine in BASIC:

```
5100  REM  SCREEN SCROLLER
5110  REM  INPUT VARIABLES - SS
5120  REM  OUTPUT VARIABLES -
5130  REM  INTERNAL VARIABLES - I9
5140  VTAB 24: SPEED= 100
5150  FOR I9 = 1 TO SS: PRINT : NEXT I9
5160  SPEED= 255: RETURN
```

How to call routine from program: The input SS (Screen Scroller) controls the number of lines to be scrolled up on the screen. The Applesoft command SPEED is used to slow the process. This routine can be called from any loction in the main program.

Sample program usage:

```
25000  REM  SCREEN FORMAT
25010  HOME
25020 TS = "SCREEN FORMAT": GOSUB 6500
25030 WL = 0:WW = 40:WT = 7:WB = 24: GOSUB 5000
25100 TS = "PAGE 1": GOSUB 6200
25110  VTAB 20:TS = "THIS IS A DEMONSTRATION OF": GOSUB 6200
25120 TS = "A SCROLLING SCREEN DISPLAY!": GOSUB 6200
25130 TD = 2000: GOSUB 8500
25140 SS = 14: GOSUB 5100
25150  VTAB 8:TS = "PAGE 2": GOSUB 6200
25160  VTAB 20:TS = "THIS IS A DEMONSTRATION OF": GOSUB 6200
25170 TS = "A WHOLE SCREEN ORIENTATION!": GOSUB 6200
25180  POKE  - 16368,0:VP = 23: GOSUB 8000
25190 TS = "PAGE 3": GOSUB 6200
25200 VP = 23: GOSUB 8000
```

Screen Oriented Format

Description of purpose and use: Presents information to the user one SCREEN or page at a time.

How to check a program for routine: Programs using this technique present information one page at a time. Processing can be delayed by a time loop (FOR-NEXT). Commonly INPUT or (GET or INKEY) statements are used to cause the computer to waite for a key(s) to be pressed. Each page is usually starting on a new screen (by using the HOME or CLS command).

How to program routine in BASIC:

```
25150  VTAB 8:T$ = "PAGE 2": GOSUB 6200
25160  VTAB 20:T$ = "THIS IS A DEMONSTRATION OF": GOSUB 6200
25170 T$ = "A WHOLE SCREEN ORIENTATION!": GOSUB 6200
25180  POKE  - 16368,0:VP = 23: GOSUB 8000
25190 T$ = "PAGE 3": GOSUB 6200
25200 VP = 23: GOSUB 8000
```

How to call routine from program: The screen orientation desired is produced by the GETS and HOME commands found in the USER SELF-PACER subroutine starting on line 8000. The Vertical Position (VP) of the user prompt must be sent to the subroutine. The POKE -16368,0, for the Apple II, clears the keyboard buffer (to remove the effect of the user pressing extra keys) prior to calling the subroutine.

Window Poker

Description of purpose and use: Allows programmer to change monitor screen window to create special effects or to protect on-screen copy.

How to check a program for routine: If the screen window dimensions change during a program or text is displayed in the same location for several sequential screens the program is using some POKE statement to alter the computers default screen dimensions.

How to program routine in BASIC:

```
5000  REM  WINDOW POKER
5010  REM  INPUT VARIABLES - WL, WW, WT, WB
5020  REM  OUTPUT VARIABLES -
5030  REM  INTERNAL VARIABLES -
5040  POKE 32,WL: REM  WINDOW LEFTMOST POSITION
5050  POKE 33,WW: REM  WINDOW WIDTH
5060  POKE 34,WT: REM  WINDOW TOP MARGIN
5070  POKE 35,WB: REM  WINDOW BOTTOM MARGIN
5080  CALL  - 936: REM  CLEAR WINDOW/CURSOR TO TOP LEFT
5090  RETURN
```

How to call routine from program: Main program must send values for WL (Window top Left position),WW (Window Width), WT (Window Top position), and WB (Window Bottom position). Values can be: WL (0-39), WW (1-40), WT (0-23), and WB (0-24) for the APPLE II.

Sample program usage:

```
25300  REM  WINDOW POKING DEMO
25310  FOR J1 = 1 TO 15
25320  FOR J = 1 TO 40: PRINT "X";: NEXT J: NEXT J1
25330 WL = 3:WW = 3:WT = 9:WB = 20: GOSUB 5000
25340 WL = 6:WW = 30:WT = 11:WB = 18: GOSUB 5000
25350 WL = 34:WW = 3:WT = 9:WB = 20: GOSUB 5000
25360  POKE 32,0: POKE 33,40: POKE 34,0: POKE 35,24
25370  VTAB 15:TS = "  THIS IS A WINDOW DEMONSTRATION":
GOSUB 6200
25380 TD = 1000: GOSUB 8500
25390 WL = 3:WW = 34:WT = 11:WB = 18: GOSUB 5000
25400  LIST 25000 - 25430
25410  POKE 32,0: POKE 33,40: POKE 34,0: POKE 35,24
25420 VP = 23: GOSUB 8000
25430  HOME : GOTO 13000
```

Print Using For Numbers

Description of purpose and use: This routine allows number output to be placed in columns and lined-up with respect to a decimal point.

How to check a program for routine: Some microcomputer versions of BASIC have a built in PRINT USING routine. Programmers must write their own routine for those versions that do not include this aid. If a program does not line up the decimal points when numbers are placed in columns PRINT USING is not being used.

How to program routine in BASIC:

```
6000  REM  PRINT USING FOR NUMBERS
6010  REM  INPUT VARIABLES - N, DC, PR
6020  REM  OUTPUT VARIABLES - NN$
6030  REM  INTERNAL VARIABLES - I9, P9, P8, N9, N9$, C9
6040 P9 = (10 ^ PR):P8 = (1 / (10 ^ (PR + 1)) * 5)
6050 N9 =  INT ((N + P8) * P9) / P9
6060 N9$ =  STR$ (N9):C9 = 0
6070  IF  LEFT$ (N9$,1) = "." THEN 6120
6080  FOR I9 = 1 TO  LEN (N9$)
6090  IF  MID$ (N9$,I9,1) = "." THEN C9 = I9 - 1
6100  NEXT I9
6110  IF C9 = 0 THEN C9 =  LEN (N9$):N9$ = N9$ + ".": FOR I9
= 1 TO PR:N9$ = N9$ + "0": NEXT I9
6120  HTAB (DC - C9)
6130  IF  LEN (N9$) - C9 > PR THEN 6150
6140  FOR I9 = 1 TO ( ABS ( LEN (N9$) - C9 - PR - 1)):N9$ =
N9$ + "0": NEXT I9
6150 NN$ = N9$
6160  RETURN
```

How to call routine from program: The input variables N (Number), DC (Decimal Column location), and PR (desired Places Right of decimal) must be passed from the main program. The routine reforms the number to include the decimal point and number as a $tring. The output variable NN$ (New Number $tring) is passed back to the main program.

Sample program usage:

```
26100  REM  NUMERIC PRINT USING
26110  HOME
26120  FOR J = 1 TO 5
26130 P1 = (10 ^ ( LEN ( STR$ (PR)) - 2))
26140 X =  INT ( RND (1) * 100) + 1:Y =  INT ( RND (1) * 10)
+ 1
26150 Z = X / Y
26160 DC = 5:PR = 4
26170 N = X: GOSUB 6000
26180  PRINT NN$;
26190 DC = 20
```

```
26200 N = Y: GOSUB 6000
26210  PRINT NN$;
26220 DC = 30
26230 N = Z: GOSUB 6000
26240  PRINT NN$
26250  NEXT J
26260 VP = 23: GOSUB 8000
```

Centering Text

Description of purpose and use: This subroutine facilitates the centering of text strings to meet the width requirements of the screen or printer used for program output or display.

How to check a program for routine: If such a routine is not used the programmer will have to pack a PRINT statement with spaces or individually TAB each printed line to get the desired centering.

How to program routine in BASIC:

```
6200  REM  CENTERING TEXT
6210  REM  INPUT VARIABLES - T$
6220  REM  OUTPUT VARIABLES -
6230  REM  INTERNAL VARIABLES - CL
6240 CL = 40: REM    # OF SCREEN COLUMNS
6250  HTAB (((CL -  LEN (T$)) / 2) + 1)
6260  PRINT T$
6270  RETURN
```

How to call routine from program: The desired message to be centered must be defined as T$ (Text String) in the main program. The subroutine's internal variable CL (Center Location) can be changed to facilitate different width output devises (ie. the Apple monitor display is 40 characters wide and the TRS-80 display is 64 characters wide).

Sample program usage:

```
26300  REM  CENTERING TEXT
26310  HOME
26320 T$ = "THESE": GOSUB 6200
26330 T$ = "LINES ARE": GOSUB 6200
26340 T$ = "CENTERED WITH": GOSUB 6200
26350 T$ = "THE SUBROUTINE IN": GOSUB 6200
26360 T$ = "LINES 6200 THROUGH 6260": GOSUB 6200
26370 T$ = "INPUT IS GIVEN TO THE ROUTINE": GOSUB 6200
26380 T$ = "AS THE STRING VARIABLE T$ OUTPUT IS": GOSUB 6200
26390 T$ = "PLACED": GOSUB 6200
26400 T$ = "IN THE": GOSUB 6200
26410 T$ = "CENTER": GOSUB 6200
26420 VP = 23: GOSUB 8000
```

Word Wrap

Description of purpose and use: This routine allows programmer to disregard the character width restrictions of the computers monitor when developing PRINT statements. This method will WRAP excess portions of a display line on the blank space closest to but under the width specified.

How to check a program for routine: Without the use of such a routine, the programmer would have to count the number of characters in each PRINT statement to avoid WRAPPING in the middle of a word.

How to program routine in BASIC:

```
6300  REM  WORD WRAP
6310  REM  INPUT VARIABLES - T$, SC
6320  REM  OUTPUT VARIABLES -
6330  REM  INTERNAL VARIABLES - CL, T9$, C9, L9, I9
6340 CL = 40: REM  SCREEN WIDTH IN COLUMNS
6350  IF T$ = "STOP" THEN  PRINT T9$:T9$ = "": PRINT :
RETURN
6360 T$ = T9$ + T$
6370 C9 = 0
6380 T9$ =  LEFT$ (T$,CL - C9)
6390  IF  LEN (T9$) =  LEN (T$) THEN  RETURN
6400  IF  RIGHT$ (T9$,1) = " " THEN 6430
6410 C9 = C9 + 1
6420  GOTO 6380
6430  PRINT  LEFT$ (T9$,CL - 1)
6440  IF SC > 1 THEN  FOR I9 = 1 TO SC - 1: PRINT : NEXT I9
6450 L9 =  LEN (T$) -  LEN (T9$)
6460  IF L9 = 0 THEN  RETURN
6470 T$ =  RIGHT$ (T$,L9)
6480  GOTO 6370
```

How to call routine from program: To make use of this subroutine the main program must supply the input variable T$ (Text String). The input variable SC (SpaCing) will control the number of spaces between lines for that section or paragraph (ie. single or double space). After each desired string is identified in the main program and sent to the routine with a GOSUB, the next statement should be T$="STOP":GOSUB 6300. The T$="STOP" signals the end of the printing process.

Sample program usage:

```
26500  REM  WORD WRAP
26510  HOME
26520 T$ = "    THIS IS A SAMPLE PARAGRAPH SHOWING HOW WORD
WRAP WORKS.": GOSUB 6300
26530 T$ = "  WORD WRAP IS A ROUTINE THAT GETS RID OF THE
NASTY MESS CAUSED BY SPLITTING WORDS AT THE END OF LINES.":
GOSUB 6300
```

```
26540 T$ = "  ALL TOO OFTEN COMPUTER FORCED SPLITS DON'T
OCCUR IN BETWEEN SYLLABLES.": GOSUB 6300
26550 T$ = "STOP": GOSUB 6300
26560 T$ = "     WITH WORD WRAP LINES CAN ONLY END ON SPACES
NOT IN THE MIDDLE OF A WORD.": GOSUB 6300
26570 T$ = "  THIS CAUSES THE TEXT PRESENTATION TO BE MUCH
MORE READABLE.": GOSUB 6300
26580 T$ = "STOP": GOSUB 6300
26590 VP = 23: GOSUB 8000
```

Putting Text In Boxes

Description of purpose and use: This routine improves the visual effect of a screen by placing desired text information in a rectangle shape.

How to check a program for routine: If a program has several title screens with information in boxes some form of this routine is being used.

How to program routine in BASIC:

```
6500  REM  BOXES
6510  REM  INPUT VARIABLES - T$
6520  REM  OUTPUT VARIABLES -
6530  REM  INTERNAL VARIABLES - B9$, L9, H9, CL, I9
6540 CL = 40:B9$ = "*"
6550 L9 =  LEN (T$):L9 = L9 + 4
6560 H9 = ((CL - L9) / 2) + 1
6570  HTAB H9: FOR I9 = 1 TO L9: PRINT B9$;: NEXT I9: PRINT
6580  HTAB H9: PRINT B9$;: HTAB (H9 + L9 - 1): PRINT B9$
6590  HTAB H9: PRINT B9$;" ";T$;" ";B9$
6600  HTAB H9: PRINT B9$;: HTAB (H9 + L9 - 1): PRINT B9$
6610  HTAB H9: FOR I9 = 1 TO L9: PRINT B9$;: NEXT I9: PRINT
6620  RETURN
```

How to call routine from program: The desired message must be identified as T$ (Text $tring) is the main program. The internal variable B9$="*" defines the character to form the box. The box is automatically formed with the text inside.

Sample program usage:

```
26600  REM   BOXES
26610  VTAB 8
26620 T$ = "YOUR NAME IN LIGHTS!": GOSUB 6500
26630 TD = 500: GOSUB 8500: HOME
26640 T$ = N$: FLASH : GOSUB 6500: NORMAL
26650 VP = 23: GOSUB 8000
26660  HOME
```

Printer Interfacer

Description of purpose and use: This routine permits user to select use of the printer to produce hard copy. User can choose to skip the printing process. Routine prompts user to turn printer on and to position paper before printing.

How to check a program for routine: A program without such a routine will HANG if the printer is OFF LINE when the program wants to PRINT.

How to program routine in BASIC:

```
6700  REM  PRINTER INTERFACER
6710  REM  INPUT VARIABLES - VP
6720  REM  OUTPUT VARIABLES - A$
6730  REM  INTERNAL VARIABLES - A$
6740  PRINT : PRINT "DO YOU WANT PRINTER OUTPUT Y OR N ?":
PRINT "PRESS N IF PRINTER IS OFF LINE ->";
6750  GET A$
6760  IF A$ <  > "Y" AND A$ <  > "N" THEN 6750
6770  IF A$ = "N" THEN  PRINT : RETURN
6780  HOME : PRINT "PLEASE TURN PRINTER ON AND ADJUST PAPER"
6790 VP = 23: GOSUB 8000: REM  PACER
6800  PRINT : PRINT "PRINTING IS IN PROGRESS": PRINT
6810  PRINT  CHR$ (4);"PR#1": REM CHR$(4) IS CONTROL-D
6820  RETURN
```

How to call routine from program: A value for (VP) Vertical Position must be passed from the main program to position the printing choice prompt. The variable A$ is returned to the main program- (N) skips the printing. Routine activates printer interface (PR#1) and main program deactivates interface (PR#0) for the APPLE (use CMD"Z","OFF" for TRS-80).

Sample program usage:

```
26700  REM  PRINTER INTERFACER DEMONSTRATION
26710  HOME
26720 VP = 10: GOSUB 6700
26790  IF AS <  > "Y" THEN  GOTO 26890
26800  PRINT "THIS IS THE ITEM TO BE PRINTED
26810  PRINT "IF THE PRINTER IS ON!"
26820  PRINT  CHR$ (4);"PR# 0"
26890 VP = 23: GOSUB 8000
```

Typewriter Sound

Description of purpose and use: This routine clicks the computer's speaker during letter printing to simulate a typewriter or tickertape machine.

How to check a program for routine: If an APPLE II program gives a sound effect during text printing a routine similar to this one is being used.

How to program routine in BASIC:

```
6900  REM  TYPEWRITER SOUND
6910  REM  INPUT VARIABLES - TS,SP
6920  REM  OUTPUT VARIABLES -
6930  REM  INTERNAL VARIABLES - I9, J9
6940  SPEED= SP: FOR I9 = 1 TO  LEN (T$)
6950  IF  MID$ (T$,I9,1) = " " THEN  PRINT " ";: GOTO 6980
6960  PRINT  MID$ (T$,I9,1);
6970  FOR J9 = 1 TO 1:X1 =  PEEK ( - 16336): NEXT J9
6980  NEXT I9
6990  SPEED= 255: PRINT : RETURN
```

How to call routine from program: This routine requires the message to be assigned to T$ (Text $tring) in the main program. Also, a value for the variable SP (SPeed) is required to set the Applesoft SPEED command.

Sample program usage:

```
27000  REM  TUTORIAL STYLE
27010  HOME
27020 T$ = "TUTORIAL STYLE": GOSUB 6500
27030 WL = 0:WW = 40:WT = 7:WB = 24: GOSUB 5000
27040 SP = 150:T$ = "WE HAVE SEEN TWO TYPES OF TUTORIAL
STYLE": GOSUB 6900
27050 SP = 150:T$ = "USED IN THIS PROGRAM!": GOSUB 6900:
PRINT : PRINT
```

Menu Maker

Description of purpose and use: This routine allows the program user to make branching choices. In the application of Computer Assisted Instruction (CAI) learners can take different paths through instructional materials.

How to check a program for routine: If a program presents the user with a screen of choices to select and control further processing some form of this routine is being used either in the main program or in a subroutine.

How to program routine in BASIC:

```
7000  REM  MENU MAKER
7010  REM  INPUT VARIABLES - B9$, TL$, BL$, M$(I9),MH$, MN

7020  REM  OUTPUT VARIABLES - A$
7030  REM  INTERNAL VARIABLES - I9,J9
7040  HOME
7050  FOR I9 = 1 TO 40: PRINT B9$;: NEXT I9
7060  FOR J9 = 1 TO 19: PRINT B9$; TAB( 40);B9$;: NEXT J9
7070  FOR I9 = 1 TO 40: PRINT B9$;: NEXT I9
7080  HTAB ((21 - ( LEN (TL$) / 2) - 1)): VTAB 1: PRINT "
";TL$;" "
7090  HTAB ((21 - ( LEN (BL$) / 2) - 1)): VTAB 21: PRINT "
";BL$;" "
7100  HTAB 6: VTAB 4: PRINT MH$
7110  FOR I9 = 1 TO MN
7120  VTAB 6 + I9: HTAB 6: PRINT M$(I9)
7130  NEXT I9
7140  PRINT
7150  HTAB 6: PRINT "PLEASE ENTER YOUR CHOICE"
7160  HTAB 6: PRINT "=====>";
7170  GET A$
7180  IF  ASC (A$) < 65 OR  ASC (A$) > 64 + MN THEN 7170
7190  HOME : RETURN
```

How to call routine from program: The variable B9$, defined in the main program, is used to present a box on the screen. Then, TL$ (Top Label) and BL$ (Bottom Label) are made part of the screen box. The main program must also define: MH$ (menu screen's Main Heading), MN (Menu item Number or the number of desired items to be included in the menu), and the actual Menu items passed through the array M$(I9). The output variable A$ (Answer) is passed back to the main program in response to the user's menu choice. The main program should then branch according to A$.

Sample program usage:

```
11000  REM  TURORIAL MAIN MENU
11010  LET B9$ = "*"
11020 TL$ = "MAIN MENU":BL$ = "TUTORIAL"
11030  LET MN = 4
```

```
11040 MH$ = "THESE ARE OUR MAJOR CONCEPTS"
11050 M$(1) = "A. INPUT CONCERNS"
11060 M$(2) = "B. OUTPUT CONCERNS"
11070 M$(3) = "C. EDUCATIONAL CONCERNS"
11080 M$(4) = "D. EXIT PROGRAM"
11090  GOSUB 7000
11100  ON  ASC (A$) - 64 GOTO 12000,13000,14000,15000
```

User Self Pacer

Description of purpose and use: This routine delays program processing until user presses space bar. Press 'SPACE BAR' prompt is positioned on screen. Use of this routine allows users to read instruction at their own rate.

How to check a program for routine: If a program causes the user to press a single key to allow program to go on some form of this routine is being used (TRS-80 uses INKEY$ rather than GET).

How to program routine in BASIC:

```
8000  REM  USER SELF-PACER
8010  REM  INPUT VARIABLE - VP (DESIRED VTAB POSITION -1)
8020  REM  OUTPUT VARIABLES -
8030  REM  INTERNAL VARIABLES - VP, D9$
8040  VTAB VP: PRINT "**** PRESS 'SPACE BAR' TO CONTINUE
****";
8050  GET D9$: PRINT
8060  IF D9$ <  >  CHR$ (32) THEN 8040
8070  HOME
8080  RETURN
```

How to call routine from program: The input variable VP (Vertical Position) must be received from the main program to position the PRESS 'SPACE BAR' TO CONTINUE prompt under (or over) the current on screen text.

Sample program usage:

```
28200  REM  USER CONTROLLED
28210  HOME :T$ = "USER CONTROLLED": GOSUB 6200
28220  POKE  - 16368,0:VP = 23: GOSUB 8000: REM  THE POKE
CLEARS THE KEYBOARD BUFFER
```

User Pacer Pre-defined

Description of purpose and use: Causes program to delay through a pre-defined time factor. User normally does not have control over this time delay.

How to check a program for routine: If program processing is delayed without user control, some form of this (FOR-NEXT) loop is being used.

How to program routine in BASIC:

```
8500  REM  USER PACER PRE-DEFINED TIME DELAY
8510  REM  INPUT VARIABLES - TD (TIME DELAY)
8520  REM  OUTPUT VARIABLES -
8530  REM  INTERNAL VARIABLES - D9
8540  FOR D9 = 1 TO TD: NEXT D9
8550  RETURN
```

How to call routine from program: A value for the input variabe TD (Time Delay) must be defined in the main program and passed to the subroutine.

Sample program usage:

```
28100  REM  PROGRAMMER CONTROLLED PACING
28110 T$ = "PROGRAMMER CONTROLLED": GOSUB 6200
28120  VTAB 22:T$ = "PROGRAM IS IN A DELAY LOOP": GOSUB 6200
28130 TD = 3000: GOSUB 8500
```

Right Answer Routine

Description of purpose and use: This routine presents the user with randomly selected positive feedback. The routine includes the user's name in several prompts. A counter records the number of correct responses each time the routine is used.

How to check a program for routine: If a program uses randomly selected user positive feedback and counts the number of correct responses, some form of this routine is being used.

How to program routine in BASIC:

```
9000  REM  RIGHT ANSWER ROUTINE
9010  REM  INPUT VARIABLES - R, N$
9020  REM  OUTPUT VARIABLES - R
9030  REM  INTERNAL VARIABLES - X9
9040 R = R + 1
9050 X9 =  INT ( RND (1) * 10) + 1
9060  ON X9 GOTO
9070,9080,9090,9100,9110,9120,9130,9140,9150,9160
9070  PRINT "GOOD JOB ": RETURN
9080  PRINT "EXCELLENT ";N$: RETURN
9090  PRINT "YOU GOT IT!": RETURN
9100  PRINT "NO DOUBT ABOUT IT!": RETURN
9110  PRINT "VERY GOOD ";N$: RETURN
9120  PRINT "WONDERFUL!": RETURN
9130  PRINT "BINGO": RETURN
9140  PRINT "WHAT AN ACE!": RETURN
9150  PRINT "CORRECT": RETURN
9160  PRINT N$;" YOU ARE RIGHT": RETURN
```

How to call routine from program: This routine is usually called from the main program after a response is evaluated as correct. The input variable N$ (user's Name $tring) and R (total number Right) must be passed from the main program to include the user's name in random responses and to update the right item counter. The updated right item counter (R) is passed back to the main program.

Sample program usage:

```
29100  REM  RIGHT ANSWER
29110 T$ = "TRY TO ANSWER THE FOLLOWING QUESTIONS
CORRECTLY.": GOSUB 6300
29120 T$ = "STOP": GOSUB 6300: PRINT
29130 T$ = "TRUE OR FALSE - THE ACRONYM FOR SOFTWARE WHICH
TRANSFERS INFORMATION BETWEEN A COMPUTER AND IT'S DISK
DRIVES IS DOS.": GOSUB 6300
29140 T$ = "STOP": GOSUB 6300
29150  GET A$
29160  IF A$ <  > "T" AND A$ <  > "F" THEN  GOSUB 1000: GOTO
29150
```

```
29170  PRINT
29180 CA$ = "T": GOSUB 3100
29190  IF CR = 1 THEN  GOSUB 9000
29200  IF CR = 0 THEN  GOSUB 9200
29210 VP = 23: GOSUB 8000
```

Wrong Answer Routine

Description of purpose and use: This routine presents the user with randomly selected negative feedback after an incorrect response. The routine includes the user's name in several prompts. Two counters are used. One counter records the overall number of wrong responses each time the routine is used. The second counter is usually reset in the main program and can be used to control the number of user attempts for an item.

How to check a program for routine: If a program uses randomly selected user negative feedback, controls the number of user attempts at a question, or counts the number of wrong responses, some form of this routine is being used.

How to program routine in BASIC:

```
9200  REM  WRONG ANSWER ROUTINE
9210  REM  INPUT VARIABLES - W, C, N$
9220  REM  OUTPUT VARIABLES - W, C
9230  REM  INTERNAL VARIABLES - X9
9240 W = W + 1:C = C + 1
9250 X9 =  INT ( RND (1) * 10) + 1
9260  ON X9 GOTO
9270,9280,9290,9300,9310,9320,9330,9340,9350,9360
9270  PRINT "WRONG ";N$: RETURN
9280  PRINT "INCORRECT": RETURN
9290  PRINT "NOT RIGHT": RETURN
9300  PRINT N$;" THAT IS INCORRECT": RETURN
9310  PRINT "NOT THIS TIME ";N$: RETURN
9320  PRINT "NOT QUITE": RETURN
9330  PRINT "NOPE": RETURN
9340  PRINT "SORRY ": RETURN
9350  PRINT "MISTAKE": RETURN
9360  PRINT "OOPS": RETURN
```

How to call routine from program: This routine is usually called after an incorrect response is detected in the main program. The input variable W (Wrong answer counter) is used to keep track of the number of total incorrect responses. The input variable C (Counter) keeps track of the number of attempts for each individual item. N$ is the user's Name supplied from the main program. After W and C are updated program control returns to the main program.

Sample program usage:

```
29220  HOME
29230 T$ = "WHICH OF THE FOLLOWING DEVICES ALLOWS COMPUTERS
TO COMMUNICATE USING THE TELEPHONE SYSTEM?": GOSUB 6300
29240 T$ = "STOP": GOSUB 6300
29250  PRINT "     A. SPRITES"
29260  PRINT "     B. MODEMS"
29270  PRINT "     C. SUPRESSORS"
```

```
29280  PRINT "      D. DB-25S"
29290  PRINT "      E. HUB RINGS"
29300  PRINT
29310  GET A$
29320  IF  ASC (A$) < 65 OR  ASC (A$) > 69 THEN  GOSUB 1000:
GOTO 29310
29330 CA$ = "B": GOSUB 3100
29340  IF CR = 1 THEN  GOSUB 9000
29350  IF CR = 0 THEN  GOSUB 9200
29360 VP = 23: GOSUB 8000: IF CC = 1 THEN 29500
```

Section Feedback

Description of purpose and use: This routine tells the user the number of right and wrong responses and the percent of right and wrong responses.

How to check a program for routine: If a program offers the user summary statistics on section responses some form of this routine is being used.

How to program routine in BASIC:

```
10000  REM  SECTION FEEDBACK
10010  REM  INPUT VARIABLES - R, W, N$
10020  REM  OUTPUT VARIABLES -
10030  REM  INTERNAL VARIABLES - R9, W9
10040  HOME
10050  PRINT N$;"-": PRINT : PRINT "YOU GOT "; TAB( 20);R;
TAB( 25);"RIGHT"
10060  PRINT : PRINT  TAB( 20)W; TAB( 25);"WRONG"
10070  PRINT : PRINT "OUT OF "; TAB( 20);R + W; TAB(
25);"QUESTIONS."
10080 R9 = R / (R + W):R9 = R9 * 100:R9 = ( INT ((R9 + .005)
* 100)) / 100
10090 W9 = W / (R + W):W9 = W9 * 100:W9 = ( INT ((W9 + .005)
* 100)) / 100
10100  PRINT : PRINT "THAT IS ";R9;" CORRECT AND": PRINT :
PRINT W9;" INCORRECT."
10110  RETURN
```

How to call routine from program: This routine is usually called after a section of 'test' items presented in the main program. Use of this subroutine assumes that the routines for counting R (number of Right responses) and W (number of Wrong responses) was previously used. The internal variables R9 and W9 are used to convert the right and wrong answer counters into percentages. The user's name is added to the summary by use of the input variable N$ (Name String).

Sample program usage:

```
30100  HOME
30110  IF CC = 0 THEN  PRINT "PLEASE TRY FEEDBACK SECTION
FIRST.":VP = 23: GOSUB 8000:WT = 0: GOSUB 5000: GOTO 14000
30120  GOSUB 10000
30130  GOSUB 8000
```

Fading Prompts

Description of purpose and use: Fading Prompts is a CAI (Computer Assisted Instruction) method which reduces the amount of information given to the user through a series of question or problem screens.

How to check a program for routine: If a program presents the user with a series of question arranged in such a way that the user must supply more of his or her own information in sequential screens the fading prompt method is being used.

How to program routine in BASIC:

```
31260  REM  FADING PROMPTS
31270 T$ = "FADING PROMPT SCREEN 1": GOSUB 6200: PRINT
31280 T$ = "HOW MANY SQUARE INCHES ARE IN A BOARD 2 INCHES
WIDE AND 3 INCHES LONG?": GOSUB 6300
31290 T$ = "STOP": GOSUB 6300
31300  PRINT :T$ = "WIDTH (2) X LENGTH (3) = TOTAL (6)
SQUARE INCHES.": GOSUB 6300
31310 T$ = "STOP": GOSUB 6300: PRINT
31320  INPUT "ENTER ANSWER THEN PRESS 'RETURN'==>";A$
31330  IF A$ <  > "6" THEN  PRINT "NO":VP = 23: GOSUB 8000:
GOTO 31260
31340  PRINT "GOOD":VP = 23: GOSUB 8000
31350 T$ = "FADING PROMPT SCREEN 2": GOSUB 6200: PRINT
31360 T$ = "HOW MANY SQUARE INCHES ARE IN A BOARD 9 INCHES
WIDE AND 10 INCHES LONG?": GOSUB 6300
31370 T$ = "STOP": GOSUB 6300: PRINT
31380  INPUT "ENTER ANSWER THEN PRESS 'RETURN'==>";A$
31390  IF A$ <  > "90" THEN  PRINT "WRONG":VP = 23: GOSUB
8000: HOME : GOTO 31350
31400  PRINT "GOOD":VP = 23: GOSUB 8000
```

Help Screens

Description of purpose and use: Help screens are used to make a program more 'user friendly' by providing access to instructions or important information during program execution. This method displays the help screen when the user presses the (ESC) key in response to a question.

How to check a program for routine: A program uses HELP SCREENS if it allows the user to access instructions or other important information based on the user's needs.

How to program routine in BASIC:

```
31100  REM    HELP SCREENS
31110 A7$ = "MICROCOMPUTER":A6$ =  CHR$ (13)
31120 A5$ = A7$ + A6$
31130  PRINT "IF YOU NEED HELP PRESS";: FLASH : PRINT
"'ESC'": NORMAL
31140 A4$ = ""
31150  PRINT : PRINT "ENTER THE PASS WORD FOR THIS PROGRAM"
31160  PRINT "THEN PRESS 'RETURN'!": PRINT
31170  GET A$
31180  IF  ASC (A$) = 8 THEN A4$ =  LEFT$ (A4$,( LEN (A4$) -
1)): GOTO 31200
31190 A4$ = A4$ + A$
31200  IF  ASC (A$) = 27 THEN  GOSUB 31420: HOME : GOTO
31130
31210  IF  ASC (A$) = 8 THEN  PRINT A$;" ";
31220  PRINT A$;
31230  IF A4$ = A5$ THEN  PRINT : PRINT "THAT IS IT!":VP =
23: GOSUB 8000: GOTO 31260
31240  IF  ASC (A$) = 13 AND A4$ <  > A7$ THEN  PRINT :
PRINT "WRONG":VP = 23: GOSUB 8000: HOME :A4$ = "": GOTO
31130
31250  IF  ASC (A$) <  > 13 THEN 31170
31420  REM   HELP SCREEN FOR ABOVE
31430  HOME :T$ = "THIS IS A HELP SCREEN": GOSUB 6200
31440  PRINT : PRINT : FLASH
31450 T$ = A7$: GOSUB 6200: NORMAL
31460 VP = 23: GOSUB 8000
31470  RETURN
```

How to call routine from program: If ASC (A$)=8 the program has detected that the user has pressed the 'ESC' key (ASC=8) in response to a question (A$). The HELP SCREEN is then displayed.

Exit Routine

Description of purpose and use: Gives user a departing message, starts a new screen, and gives user RESTART instructions.

How to check a program for routine: A user friendly program will tell the user how to restart the program.

How to program routine in BASIC:

```
15000  REM  PROGRAM EXIT
15010  FOR J9 = 1 TO 23
15020  PRINT  TAB( J9 * 1.5)"BYE!": NEXT J9
15030  LET TD = 1000: GOSUB 8500: HOME
15040  PRINT "TYPE 'RUN' AND PRESS 'RETURN' TO RESTART": END
```

How to call routine from program: Program control can be passed to this routine from any location in the program. However, in a main menu driven program a user can only reach the exit routine from the MAIN MENU. Note that this routine passed a value for TD (time delay) to the time delay subroutine starting at line 8500.

Order Form

CAI Subroutine Disk

For Apple II Computers

Please enclose $10 for each disk wanted.

Name: _____

Address: _____

City: _____ State: _____ Zipcode: _____

Phone: _____

Number of disks wanted: _____

Date: _____

Mail to:

Marvin I. Sarapin

907 Essex

West Lafayette, IN 47906

(317) 463-5875

Make checks payable to Marvin I. Sarapin

For office use only:

Order taken by: _____

Order filled by: _____

Date: _____